Getting started with uClinux on adStar

Ver 1.1

April 26, 2013

Advanced Digital Chips Inc.

(www.adc.co.kr)

History

Ver 1.0

April 23, 2012	1st version released

Ver 1.1

April 26, 2013	Edit kernel download address
	Edit ramdisk

Office
22F, KeumKangPenterium IT Tower A dong, 810, Gwanyang-dong, Dongan-gu, Anyang-si ,

Gyeonggi-do, 431-060, Korea

Tel : + 82-31-463-7500

Fax : + 82-31-463-7588

Contents

1. Introduction

이 문서는 adchips adStar STK보드에서 uClinux개발에 필요한 내용을 제공하기 위해 작성되었다. EISC uClinux는 MMU가 없는 ae32k hardware 플랫폼을 지원하기 위한 Linux 운영체제이며 'FLAT'파일 포맷의 응용 프로그램을 실행가능하도록 구현되었다.

1.1. EISC uClinux Software Package.

- uClinux
  - http://www.adc.co.kr/download/uClinux/eisc_linux-2.6.29.1-uc0.tar.gz
- Toolchain
  - http://www.adc.co.kr/download/uClinux/EISC_uClinux2.6_Compiler_Source_linux.tar.gz
- Ramdisk
  - http://www.adc.co.kr/download/uClinux/ramdisk.img.gz
- U-boot
  - http://www.adc.co.kr/download/uClinux/u-boot-1012.tar.gz

1.2. Development Platform

Windows가 설치된 호스트 컴퓨터와 Linux 환경(가상머신 또는 서버)을 필요로 한다. adStar STK 보드로 커널 및 응용 프로그램을 다운로드하기 위한 툴킷은 Windows에서 사용가능하다.

1.3. Software Requirements

- RemoteManCLI.exe
  - adStar SDK내에 포함(pc-utils₩RemoteManCLI.exe)
  - http://www.adc.co.kr/download/adStar/Tools&Software/SDK/SDK_v1.2.5.zip

1.4. Hardware Requirements

- adStar STK보드
  - http://adc.co.kr/product/product3.php?cc_code=2010&cg_no=40
- E-CON toolkit
  - http://adc.co.kr/product/product3.php?cc_code=2012&cg_no=22

2. Setting up EISC uClinux Development Enviornment

아래는 Ubuntu10.04 LTS버전의 Linux 운영체제에서 EISC uClinux 컴파일러를 설치하여 크로스 개발환경을 구축하는 과정이다.

Linux호스트에서 다운받은 툴체인의 압축을 해제한다.

```
~$ tar xvfz EISC_uClinux2.6_Compiler_Source_linux.tar.gz
```

Linux환경에서 EISC uClinux 컴파일러를 설치하는 대부분의 과정은 스크립트로 작성되어져 있다. 따라서 사용자가 수정해야 할 부분은 'Makefile'파일을 열어서 설치될 경로를 지정하고 Host 컴파일러의 gcc 버전이 3.4인것을 확인하는 것이다.

먼저 압축해제된 디렉토리로 이동한 후 'Makefile'파일을 열어 'COMPILER_PREFIX'를 '/usr/local'로 수정하여 '/usr/local/ae32000-elf-uclibc-tools' 경로에 설치되도록 한다.

```
~$ cd ~/EISC_uClinux2.6_Compiler_Source_linux
~/EISC_uClinux2.6_Compiler_Source_linux $ cd AE32000C_gcc-3.4.5-v2.6.5
~/EISC_uClinux2.6_Compiler_Source_linux/AE32000C_gcc-3.4.5-v2.6.5$
vi Makefile
```

```
COMPILER_PREFIX=/usr/local
```

컴파일하기전 리눅스머신에 설치된 호스트 컴파일러의 버전을 확인한다.

```
~$ gcc -v
```

-

만약 호스트 컴파일러의 버전이 v3.4보다 높다면 다운그레이드해야만 한다. Ubuntu의 패키지 관리 프로그램을 통해 gcc-3.4를 설치한다. 먼저 패키지 저장소를 변경하기 위해 /etc/apt/source.list 파일을 백업해두고 아래 링크의 sources.list_8.04 파일을 /etc/apt/source.list 파일로 복사한 후 아래 명령을 통해 패키지 저장소를 업데이트하여 gcc-3.4를 설치한다.
http://www.adc.co.kr/download/uClinux/sources.list_8.04)

```
~$ cd /etc/apt/
/etc/apt/$ sudo cp sources.list sources.list_bak
/etc/apt/$ sudo cp /media/sf_share/sources.list_8.04 ./sources.list
/etc/apt/$ sudo apt-get update
/etc/apt/$ sudo apt-get install gcc-3.4
```

gcc명령어의 링크를 gcc-3.4로 변경한 후 gcc버전을 확인한다.

```
~$ sudo rm -f /usr/bin/gcc
~$ sudo ln -s /usr/bin/gcc-3.4 /usr/bin/gcc
~$ gcc -v
Reading specs from /usr/lib/gcc/i486-linux-gnu/3.4.6/specs
Configured with: ../src/configure -v --enable-languages=c,c++,f77,pascal --
prefix=/usr --libexecdir=/usr/lib --with-gxx-include-dir=/usr/include/c++/3.4 --
enable-shared --with-system-zlib --enable-nls --without-included-gettext --
program-suffix=-3.4 --enable-__cxa_atexit --enable-clocale=gnu --enable-
libstdcxx-debug --with-tune=pentium4 i486-linux-gnu
Thread model: posix
gcc version 3.4.6 (Ubuntu 3.4.6-6ubuntu5)
~$
```

EISC uClinux Compiler을 컴파일하기 위해 필요한 의존성 라이브러리를 설치한다. 라이브러리가 없을 경우 컴파일시 에러가 발생한다.

```
~$ sudo apt-get install bison
~$ sudo apt-get install flex
~$ sudo apt-get install gettext
```

EISC uClinux Compiler의 압축이 해제된 디렉토리로 이동후 make 명령을 통해 EISC uClinux Compiler를 설치한다.

```
~$ cd  ~/work/EISC_uClinux2.6_Compiler_Source_linux/AE32000C_gcc-3.4.5-
v2.6.5
~/work/EISC_uClinux2.6_Compiler_Source_linux/AE32000C_gcc-3.4.5-v2.6.5
$ sudo make
```

자신의 홈디렉토리에 위치한 .bashrc파일을 열어서 다음과 같이 PATH변수를 추가하여 설치된 toolchain경로를 지정한다.

```
~$ vi ~/.bashrc
PATH=/usr/local/ae32000-elf-uclibc-tools/bin:$PATH
```

변경된 PATH 환경변수를 적용시킨다.

```
~$ source ~/.bashrc
```

ae32000-elf-uclibc-gcc명령어를 실행하여 버전을 확인한다.

```
adc@adc-sys:~$ ae32000-elf-uclibc-gcc -v
Reading specs from /usr/local/ae32000-elf-uclibc-
tools/lib/gcc/ae32000-elf-uclibc/3.4.5/specs
Configured with:
/home/adc/work/EISC_uClinux2.6_Compiler_Source_linux/AE32000C_gcc-
3.4.5-v2.6.5/toolchain_build_ae32000/gcc-3.4.5-ae32000c-uclibc-
v110707/configure --prefix=/usr/local/ae32000-elf-uclibc-tools --
build=i386-pc-linux-gnu --host=i386-pc-linux-gnu --target=ae32000-
elf-uclibc --enable-languages=c,c++ --with-gxx-include-
dir=/usr/local/ae32000-elf-uclibc-tools/ae32000-elf-uclibc/include/c++
--disable-shared --disable-__cxa_atexit --enable-target-optspace --
with-gnu-ld --without-pic --disable-nls --enable-sjlj-exceptions
Thread model: single
gcc version 3.4.5 (AE32000 Compiler v2.6.5 | binutils-2.14 | gdb_insight-
6.8)
(LDI Code motion / Seperated GCCLIB / mulsi3 / Mem index
/ Floating-point optimized again / Double precision BUG Fixed)
adc@adc-sys:~$
```

3. Kernel configuration & Compile

adStar STK보드의 기본 설정파일을 가져와서 설정을 수정하도록 한다. 커널 소스의 최상위 디렉토리에 adStar 설정파일인 adstar_defconfig파일을 .config파일로 복사한 후 make 명령을 사용하여 컴파일한다. 만약 ncurses 라이브러리가 없을 경우 menuconfig명령시 에러가 발생할수 있으니 백업해두었던 패키지저장소 목록을 복구한 후 업데이트하여 ncurses-dev 라이브러리를 설치한다.

```
~$ cd /etc/apt
/etc/apt$ sudo cp sources.list_bak sources.list
/etc/apt$ sudo apt-get update
/etc/apt$ sudo apt-get install ncurses-dev
```

다운받은 커널 소스파일의 압축을 해제하여 디렉토리로 이동한다.

```
~$ tar xvfz eisc_linux-2.6.29-uc0.tar.gz
~$ cd eisc_linux-2.6.29-uc0
```

adStar STK보드의 설정파일을 복사한 후 make menuconfig 명령으로 설정을 변경하도록 한다. 기본설정파일은 'arch/ae32k/config'디렉토리에 존재한다.

```
~/eisc_linux-2.6.29-uc0$ cp arch/ae32k/configs/adstar_defconfig .config
~/eisc_linux-2.6.29-uc0$ make menuconfig
```

별도의 수정없이 기본설정파일을 사용하려면 다음과 같이 oldconfig명령을 사용한다.

```
~/eisc_linux-2.6.29-uc0$ cp arch/ae32k/configs/adstar_defconfig .config
~/eisc_linux-2.6.29-uc0$ make oldconfig
```

3.1. SystemType

다음은 make menuconfig명령을 사용하였을 경우 각 메뉴에 대해 살펴보겠다. SystemType설정은 다음과 같다.

```
SystemType--->
    AE32K system type (adchips adStar)
      :  사용하고자 하는 시스템타입을 선택한다. adchips adStar가 체크되어 있는
      지
        확인한다.
```

[*] Set dram/ramdisk base address and size

(0x20000000) DRAM Base Address

: DRAM의 시작주소를 적어준다.

(0x01000000) DRAM SIZE

: DRAM의 크기를 적어준다.

(0x20a00000) RAMDISK_BASE

: DRAM에 적재되는 램디스크의 시작주소를 적어준다.

(0x200000) RAMDISK_SIZE

: DRAM에 적재되는 램디스크의 크기를 적어준다.퍄

Uncached DMA region (Enable 1M DMA region)

: cache disable 영역의 크기를 설정한다.

[*]Install vectors to begging of RAM

: DRAM에 벡터테이블을 재설정한다.

(0x2010F000)Location of the interrupt vector table

: DRAM주소 일 경우 커널이미지의 vectors섹션주소로 설정되므로 기본값으로 둔다.

3.2. Kernel Features

Kernel Features설정은 다음과 같다.

Kernel Features--->

uClinux kernel load address(0x20000000)

: 커널이 실행될 주소를 적어준다. EISC uClniux kernel은 DRAM의 시작주소에 커널이 적재되어 실행되므로 RAM의 시작주소와 같다.

3.3. Boot Options

Boot options설정은 다음과 같다.

Boot options--->

Default kernel command string

(rootfstype=ext2 root=/dev/ram0 console=ttyS0,115200)

:커널부팅시 사용될 command_string을 이곳에 적어둔다. 위와 같이 설정되면 커널 부팅 메세지와 콘솔 메세지가 출력된다..

3.4. Device Driver

3.4.1. Serial Drvier

Serial관련 설정은 다음과 같다.

Device Driver--->Character devices--->Serial drivers

[*]adStar on-chip serial port support

: adStar uart serial driver를 사용한다.

[*]Support for console on adStar serial port

: uart console driver를 사용한다.

[*]adStar Initial serial console speed

: uart baudrate를 설정한다. adStar의 경우 115200을 선택한다.

3.4.2. Nand

Nand관련 설정은 다음과 같다.

Device Driver--->

<*>Memory Technology Devicce(MTD support --->

[*]MTD Partitioning support

: flash에 2개이상의 파티션을 나눌 수 있도록 해준다.

<*> NAND Devicce Support --->

[*]NAND Flash device on adStar

: adStar의 NAND Flash device driver를 사용한다

Nand를 설정하였을 경우 SD card 설정을 해제한다. adStar의 경우 GPIO PinMuxing을 통해 둘 중 하나만 사용가능하다.

3.4.3. SD card

SDcard관련 설정은 다음과 같다.

Device Driver--->MMC/SD/SDIO card support--->

[*]MMC block device driver

: SD card의 파일시스템을 마운트할 수 있도록 해준다.

[*]adchips adStar SD/MMC Card interface support

: adStar Sdcard device driver를 사용한다.

SD card를 설정하였을 경우 Nand 설정을 해제한다. adStar의 경우 GPIO PinMuxing을 통해 둘 중 하나만 사용가능하다.

3.4.4. LCD

LCD관련 설정은 다음과 같다.

Device Driver--->Graphics support--->
<*>Support for frame buffer devices--->
<*>adStar LCD framebuffer support
: adStar LCD framebuffer driver를 사용한다.

(LOGO)
[*]Bootup logo--->
: 부팅시 LCD화면 상단에 로고 이미지를 출력한다. 하위 메뉴중 하나의 로고를 선택한다.

(CONSOLE)
Device Driver--->Character devices--->
[*]Virtual Terminal
[*]Support for console on virtual terminal
Device Driver--->Graphics support---> Console display driver support --->
<*>Framebuffer Console support
: LCD를 통해 printk메시지 출력이 가능하다.
[*]Map the console to the primary display device
: 첫번째 display장치에 콘솔이 매핑될 수 있도록 설정해둔다.
Boot options --->
(rootfstype=ext2 root=/dev/ram0 console=ttyS0,115200 console=tty0) Default kernel command string

Boot options에서 'console=tty0 '을 추가할 경우 프레임버퍼를 통해 콘솔메세지의 출력이 가능하다. 따라서 위와 같이 설정하였을 경우 콘솔메세지가 프레임버퍼와 UART로 출력이 가능하다.

3.4.5. TouchScreen & I2C

TuchScreen과 I2C관련 설정은 다음과 같다.

Devicce Driver--->Input device support
[*]Touchscreens--->
<*>adStar AK4183 Touchscreen Interface(NEW)
.Devicce Driver--->
<*>I2C support--->
<*>I2C device interface
: 사용자 공간의 응용프로그램에서 /dev디렉토리의 장치파일을 통해 I2C버스를 접근하고자 할 경우 선택한다.

I2C Hardware Bus support --->

<*>adchips adStar I2C Two-Wire interface(TWI)(NEW)

: adstar I2C device driver를 사용한다. Touchscreen을 사용할 경우 i2c를 통해 통신이 이루어진다.

3.4.6. USB Device(Bulk transfer)

USB Device 설정은 다음과 같다.

Devicce Driver--->

[*]USB support --->

<*>adStar USB Device support(bulk)

설정이 완료되면 저장한 후 종료한다.

4. Compile & Run

설정을 끝마쳤다면 make명령을 통해 커널을 컴파일하여 linux.bin 바이너리 파일을 생성한다.

```
~/eisc_linux-2.6.29-uc0$ make
 ....
LD       linux
SYSMAP   System.map
OBJCOPY linux.bin
OBJCOPY arch/ae32k/boot/linux.bin
Building modules, stage 2.
MODPOST 0 modules
[~/eisc_linux-2.6.29-uc0]$
```

컴파일이 성공적으로 끝나면 linux.bin 이라는 이름을 가진 바이너리 파일을 adStar SDK Reference Manual의 3.1.2 bootloader mode를 참고하여 Remote Communicatio Mode에서 다운로드 및 실행가능하다.
(http://www.adc.co.kr/download/adStar/Documents/SDK/adStar_SDK_Reference_Manual_ko_v1.2.pdf)

adStar SDK의 부트로더를 통해 Remote Commucation Mode로 진입하였다면 console창에 다음과 같이 명령어를 입력하여 커널이미지와 램디스크를 DRAM에 Write하고 실행가능하다.

```
C:₩work> RemoteManCLI.exe -target adstar -fw 0x20000000 linux.bin -fw
0x20a00000 ramdisk.img.gz -run 0x20000000 -exit
```

root로 로그인 시 비밀번호를 묻지않고 쉘 프롬프트가 출력된다.

```
ae32k login: root
welcome to

         _________
  __  __/ ____/ (_)___  __  ____  __
 / / / / /   / / / __ \/ / / / |/_/
/ /_/ / /___/ / / / / / /_/ />  <
|__,_/|____/_/_/_/ /_/|__,_/_/ |_|
  for
      ___     _____________   __ __
     /   |   / ____/__  /__ \ |/ //_/
    / /| |  / __/   /_ <__/ // ,<
   / ___ |/ /___ ___/ / __// /| |
  /_/  |_/_____//____/____/_/ |_|
             (http://www.adc.co.kr)



BusyBox v1.19.2 (2012-04-13 14:13:55 KST) hush - the humble shell

~ #
```

U-Boot 부트로더에서 커널을 부팅하려면 'arch/ae32k/boot'디렉토리에 생성된 'uImage'파일을 사용한다. 'uImage' 파일은 u-boot헤더와 'linux.gz'압축파일로 구성된다. U-Boot 부트로더를 통해 Remote Communication Mode로 진입하려면 이 문서의 7.1, 7.3, 7.4단계를 진행하고 커맨드 모드에서 'rsp'명령을 사용한 후 Windows 콘솔창에서 위의 명령을 실행한다. 생성된 'uImage' 파일은 '7 .Booting Linux kernel using U-Boot' 단계에서 'bootm'명령을 사용하여 실행한다.

.

5. Test cases

5.1. Nand

adStar STK보드에서 NAND모듈을 실행하면 다음과 같은 메시지가 출력된다.

```
[42949374.540000] NAND device: Manufacturer ID: 0xec, Chip ID: 0xf1 (Samsung
NAND 128MiB 3.3V 8-bit)
[42949374.550000] Scanning device for bad blocks
[42949374.560000] Bad eraseblock 8 at 0x000000100000
[42949374.570000] Bad eraseblock 8 at 0x000000120000
[42949374.580000] Bad eraseblock 8 at 0x000000140000
[42949374.590000] Bad eraseblock 8 at 0x000000160000
[42949374.600000] Bad eraseblock 8 at 0x000000180000
[42949374.610000] Bad eraseblock 8 at 0x0000001a0000
```

Spare영역에서 첫번째, 두번째 바이트를 확인하여 패턴이 일치하지 않으면 베드섹터로 판단한다. 이 에러메세지를 발생하지 않기 위해서 해당 영역을 erase한다. driver/mtd/tests 디렉토리에 MTD메모리를 erase할 수 있는 코드(mtd_erase.c)를 작성해두었으므로 이 코드를 컴파일하여 모듈을 실행해보자. 해당 파일을 모듈로 컴파일 하기 위해서는 커널 설정을 다음과 같이 하면 된다.

```
Device Drivers--->Memory Technology Device(MTD) support--->
  <M>MTD tests support

Device Driver--->Memory Technology Devicce(MTD support --->
  []MTD Partitioning support
  :MTD Partitioning support의 설정을 해제한다.
```

커널 컴파일 후 생성된 mtderase.ko 오브젝트 파일을 램디스크 이미지에 넣고 커널을 실행하여 모듈을 적재해본다.

```
~$ cp /media/sf_share/ramdisk.img.gz .
~$ gunzip ramdisk.img.gz
~$ sudo mount -t ext2 ramdisk.img /mnt/img -o loop
~$ sudo cp eisc_linux-2.6.29-uc0/drivers/mtd/tests/mtd_erase.ko /mnt/img/root
~$ sudo umount /mnt/img
~$ gzip ramdisk.img
```

.

```
~ # insmod mtd_erase.ko
[    61.980000]
```

```
[    61.990000]=============================================
[    62.000000] mtd_bbterase: MTD device: 0
[    62.010000] mtd_bbterase: MTD device size 134217728, eraseblock size
131072, page size 2048, count of eraseblocks 1024, pages per eraseblock 64,
OOB size 64
[    62.020000] mtd_bbterase: erasing all blocks
[    63.860000]=============================================
~ #
```

5.2. SDcard

EISC uClinux 시스템에서 sd카드를 인식하기 위해서 파티션 테이블을 새로 만들어줘야 한다. 해당 시스템에서는 sd카드의 파티션을 잡기 위해서 msdos파티션 테이블을 사용하기 때문에 busybox의 fdisk명령과 mkfs.vfat명령을 사용하여 파티션을 새로 할당하고 MS-DOS파일시스템을 생성한다.

fdisk명령을 다음과 같이 실행한다. Sd카드의 장치파일은 '/dev/mmcblk0'이다.

```
~ # fdisk /dev/mmcblk0

The number of cylinders for this disk is set to 15568.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help):
```

'o'명령을 입력하여 DOS partition table을 생성한다.

```
Command (m for help): o
Building a new DOS disklabel. Changes will remain in memory only,
until you decide to write them. After that the previous content
won't be recoverable.

The number of cylinders for this disk is set to 15568.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
```

1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
(e.g., DOS FDISK, OS/2 FDISK)

Command (m for help):

'p'명령을 입력하여 현재 파티션테이블을 출력해본다.

Command (m for help): **p**

Disk /dev/mmcblk0: 510 MB, 510132224 bytes
4 heads, 16 sectors/track, 15568 cylinders
Units = cylinders of 64 * 512 = 32768 bytes

Device	Boot	Start	End	Blocks	Id	System

Command (m for help):

만약 파티션이 존재한다면 'd'명령을 사용하여 삭제 후 새로 추가한다. 파티션을 추가하는 명령은 'n'명령이다. 아래는 sd카드메모리 전체를 파티션 1번으로 잡는다.

Command (m for help): **n**
Command action
e extended
p primary partition (1-4)
p
Partition number (1-4): **1**
First cylinder (1-15568, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-15568, default 15568):
Using default value 15568

Command (m for help): **p**

Disk /dev/mmcblk0: 510 MB, 510132224 bytes
4 heads, 16 sectors/track, 15568 cylinders
Units = cylinders of 64 * 512 = 32768 bytes

```
          Device Boot      Start         End      Blocks   Id System
/dev/mmcblk0p1               1       15568      498168   83 Linux

Command (m for help):
```

파티션 추가 후 파티션테이블을 출력해보면 System이 'Linux'로 되어있으므로 't'명령을 입력하여System 타입을 'Win95 FAT32'로 변경한다.

```
Command (m for help): t
Selected partition 1
Hex code (type L to list codes): b
Changed system type of partition 1 to b (Win95 FAT32)

Command (m for help): p

Disk /dev/mmcblk0: 510 MB, 510132224 bytes
4 heads, 16 sectors/track, 15568 cylinders
Units = cylinders of 64 * 512 = 32768 bytes

          Device Boot      Start         End      Blocks   Id System
/dev/mmcblk0p1               1       15568      498168    b Win95 FAT32
```

파티션의 시스템타입이 Win95로 변경되었음을 확인하고 'w명령을 입력하여 파티션정보를 sd카드에 저장한다.

```
Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table
[   199.930000]  mmcblk0: p1
~ #
```

mkfs.vfat명령을 사용하여 MS-DOS파일시스템을 생성한다.

```
~ # mkfs.vfat /dev/mmcblk0p1
~ #
```

SD카드를 제거하고 다시 인식시킨다. 아래와 같이 메모리 크기와 'mmcblk0: p1'이라고 출력되면

파티션이 정상적으로 잡힌 것이다.

```
~ # [ 1332.120000] mmc0: card b368 removed
~ # [ 1338.090000] mmc0: host does not support reading read-only switch.
assuming write-enable.
[ 1338.100000] mmc0: new SD card at address b368
[ 1338.120000] mmcblk0: mmc0:b368 SD      486 MiB
[ 1338.130000]  mmcblk0: p1
~ #
```

Mount명령을 사용하여 /dev/mmcblk0p1 장치파일을 /mnt/sd 디렉토리로 마운트해보자.

```
~ # mount /dev/mmcblk0p1 /mnt/sd
~ #
```

현재 sdcard는 fat 파일시스템으로 포맷되어 있다. 만약 위와 같이 mount하였으나 다음과 같은 에러메세지가 출력될 경우가 있다.

```
~#mount /dev/mmcblk0p1 /mnt/sd
mount: mounting /dev/mmcblk0p1 on /mnt/sd/ failed: Invalid argument
```

이와 같은 경우 커널이 해당 파일시스템을 지원하지 않아서 발생할 수 있는 문제이므로 커널설정을 확인한다..

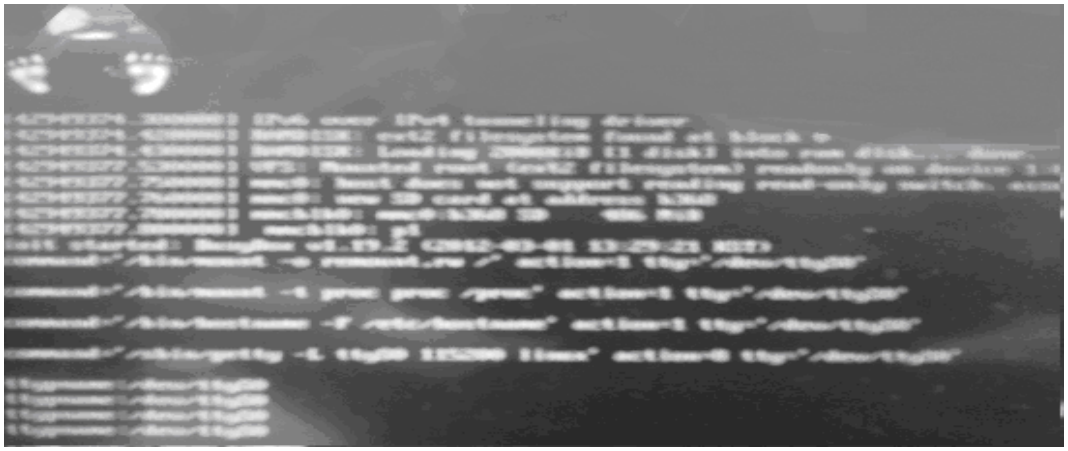
```
File System--->DOS/FAT/NT Filesystem
<*>MSDOS fs support
<*>VFAT (Windows-95) fs support
(949) Default codepage for FAT(NEW
(iso8859-1) Default iocharset for FAT(NEW

File System--->Native language support--->
(iso8859-1) Default NLS Option
<*> Korean charset(CP949, EUC-KR)
<*>NLS ISO 8859-1(Latin1; Western European Languages)

File System-->Partition Types -->
 [*]Advanced partition selection
  [*]PC BIOS(MSDOS partition tables) support
```

5.3. LCD

lcd설정을 하였을 경우 커널 부팅시 다음과 같이 LCD에 이미지와 부팅메세지가 출력됨을 확인할 수 있다.



5.4. Touchscreen & I2C

Touchscreen과 I2C설정을 활성화한 후 커널부팅이 되면 LCD화면에 터치입력이 가능하다. 입력시 다음과 같이 좌표가 출력됨을 확인할 수 있다.

```
[42949387.530000] ak4183 ts: x: 476, y:152
[42949387.560000] ak4183 ts: x: 609, y:210
[42949387.760000] ak4183 ts: x: 629, y:210
[42949388.780000] ak4183 ts: x: 271, y:156
[42949388.280000] ak4183 ts: x: 322, y:212
[42949388.550000] ak4183 ts: x: 506, y:228
```

5.5. USB device(Bulk transfer)

윈도우 호스트의 RemoeManCLI.exe와 uClinux의 rsp프로그램을 통해서 바이너리파일을 램디스크로 다운로드할 수 있다. uClinux가 부팅되어 쉘 프롬프트가 출력되면 rsp프로그램을 첫번째 인자로 다운로드받은 파일의 이름을 적어주고 실행한다. 다음은 sdcard device driver를 모듈로 컴파일하여 생성된 adstarmci.ko파일을 rsp프로그램을 통해 다운로드한다. 이후 윈도우 호스트의 콘솔창에서 RemoeManCLI.exe 프로그램을 실행한다. 다운로드된 바이너리는 파일로 저장되기 때문에 주소는 의미가 없으므로 fw옵션의 주소번지는 0으로 한다

<adStar Target>

```
~ # ls
```

```
~ # rsp adstarmci.ko
~ # insmod adstarmci.ko
[   185.560000] adstar-mci adstar-mci: initialisation done.
~ # [   185.750000] mmc0: host does not support reading read-only switch.
assuming write-enable.
[   185.760000] mmc0: new SD card at address b368
[   185.770000] mmcblk0: mmc0:b368 SD      486 MiB
[   185.790000]  mmcblk0: p1
~ #
```

<Windows Host>

```
C:₩work>RemoteManCLI.exe -target adstar -fw 0x0 adstarmci.ko -q
RemoteManager(version 100)
Advanced Digital Chips Inc.
This program works with bootloader.
Target is connected
adStar
Memory information
Flash : Sector(0x1000 byte), Total size(0x80000 byte)
RAM : 0x1000000 byte
start write(19256 bytes)...
          100(19256/19256)
write :0.294493 sec
65386 bytes/sec
```

6. Compiling & Running an Application

EISC uClinux 시스템에서는 elf2flt툴과 bFLT flat file 포맷을 사용하여 실행가능한 바이너리 파일을 생성한다. 따라서 다음과 같은 Makefile을 통해 응용 프로그램을 컴파일하여 실행할 수 있다.

<Makefile>

```
COMPILER_PREFIX=/usr/local/ae32000-elf-uclibc-tools
CROSS = ae32000-elf-uclibc-
CC     = $(CROSS)gcc
ELF2FLT     = $(CROSS)elf2flt

CFLAGS += -Dlinux -D__linux__ -Dunix -D__unix__ -D__uClinux__ -DEMBED
LDFLAGS = -r -Xlinker -T$(COMPILER_PREFIX)/lib/ae32000-elf2flt.ld
```

```
TRG = test
ALLOBJS = test.o
ALLSRCS = $(ALLOBJS:.o=.c)
all: $(TRG)
$(TRG) : $(ALLOBJS)
        $(CC) $(LDFLAGS) $(ALLOBJS) -o $@.elf
        $(ELF2FLT) $(TRG).elf
        mv $(TRG).elf.bflt $(TRG)

%o : %c
        $(CC) -c $(CFLAGS) $< -o $@

clean:
        rm -f *.o *.elf *.bflt $(TRG)
```

<source>

```
#include <stdio.h>

int main()
{
    printf("Hello World!!!₩n");
    return 0;
}
```

위 Makefile의 내용에서 COMPILER_PREFIX변수를 ae32000컴파일러의 경로로 수정한다. 생성될 바이너리 파일이름을 TRG변수에 추가하고 생성될 오브젝트 파일이름을 ALLOBJS변수에 추가하여 make명령을 실행하면 컴파일 된다.

생성된 hello 실행파일을 rsp프로그램을 통해 다운로드 후 실행권한을 부여한 뒤에 실행해본다.

<adStar Target>

```
~ #rsp hello
~ #chmod u+x hello
~ #./hello
Hello World!!!
```

<Windows Host>

```
#RemoteManCLI.exe -target adstar -fw 0 hello
```

7. Booting Linux kernel using U-Boot

NAND 플래쉬메모리에 커널과 램디스크 이미지를 Write해두고 adStar STK보드의 전원이 켜지면 U-Boot Command에 의해 커널이 실행된다.

7.1. Compile U-Boot

U-Boot의 압축이 풀린 최상위 디렉토리에서 소스를 컴파일하여 u-boot.bin 바이너리 파일을 생성한다.

<Linux Host>

```
$ make mrproper
$ make adstar_config
$ make
```

7.2. Make uImage

U-Boot에서 커널을 부팅하기 위해 사용하는 커널 이미지는 'uImage' 파일이다. 'tools/'디렉토리의 mkimage실행파일을 '/usr/local/bin'디렉토리로 복사한후 커널소스의 최상위디렉토리에서 'make uImage'명령을 실행하면 U-Boot에서 실행가능한 이미지파일이 생성된다.

<Linux Host>

```
$ sudo tools/mkimage /usr/local/bin
$ cd ../eisc_linux-2.6.29-uc0
$ make uImage
…
  UIMAGE   arch/ae32k/boot/uImage
Image Name:    Linux-2.6.29-uc0
Created:       Tue May 22 16:32:31 2012
Image Type:    AE32K Linux Kernel Image (gzip compressed)
Data Size:     1166507 Bytes = 1139.17 kB = 1.11 MB
Load Address: 20000000
Entry Point:   20000000
Kernel: arch/ae32k/boot/uImage is ready
$
```

7.3. Writing U-Boot image to Serial Flash

생성된 u-boot.bin 바이너리 파일을 E-Con툴을 사용하여 Serial Flash로 Write한다.

<Windows Host>

C:₩work>**EConMan.exe -target adstar -sysinit**

EConMan Version : 1.2.5

(c) 2011 Advanced Digital Chips Inc.

.....

EConMan('q' to exit) > **ffw 0x0 u-boot.bin**

플래쉬 메모리를 지웁니다.

Sectors : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,

20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32,

플래쉬 메모리를 지웠습니다.

파일 다운로드를 시작합니다.

100% (131296/131296)

flash programming time:1.921521ms

1.9 sec for 128Kbyte, 68329 bytes/sec

데이타 검증 시작.

100% (131296/131296)

데이타 검증이 성공하였습니다.

7.4. Executing the U-boot

개발 보드를 리셋시킨 후 시리얼 터미널 프로그램(115200bps)에서 키입력시 U-boot 가 명령모드로 실행되고 키입력이 없으면 default boot command가 실행된다. 아래 그림은 adStar 보드에서 키입력시 U-boot 가 실행된 화면이다.

U-Boot 2010.12 (Apr 24 2012 - 11:14:25)

BOARD: ADSTAR

DRAM: 16 MiB

FLASH: 512 KiB

NAND: 128 MiB

In: serial

Out: serial

Err: serial

Hit any key to stop autoboot: 0

adStar #

('Warning: Bad CRC, using default environment.' 경고 메시지가 나타나는 경우는 u-boot환경변수가 flash메모리에 저장된 CRC값과 다른경우에 출력되니 'saveenv'명령을 사용하여 u-boot환경변수를 flash메모리에 재저장한다.)

```
adStar # saveenv
Saving Environment to Flash...
Un-Protected 1 sectors
Erasing Flash...70000
Erased 1 sectors
Writing to Flash... done
Protected 1 sectors
```

현재 설정된 환경변수는 다음과 같다.

```
adStar # printenv
baudrate=115200
bootcmd1=nand read.jffs2 0x20300000 0x0 0x200000; nand read.jffs2 0x20a00000
0x200000 0x200000;bootm 0x20300000
bootdelay=3
ipaddr=0
stderr=serial
stdin=serial
stdout=serial

Environment size: 205/4092 bytes
```

7.5. Writing kernel and ramdisk image to NAND Flash

Windows Host에서 RemoteManCli.exe프로그램을 사용하여 DRAM에 커널과 램디스크 이미지를 Write한 후에 adStar Target에서 nand명령을 사용하여 DRAM에 있는 이미지를 NAND 플래쉬메모리에 Write한다. uImage파일은 U-boot헤더와 커널 binary 파일이 압축되어 있는 구조이므로 0x20300000주소에 Write해두면 0x20000000 주소에 압축을 푼뒤 실행하게 된다.

adStar Target에서 rsp명령을 실행하여 Remote Communication Mode로 진입한다.

<adStar Target>

```
adStar # rsp
```

커널이미지는 0x20300000주소로 Write하고 램디스크 이미지는 0x20a00000주소로 Write한다.

<Windows Host>

```
C:₩work>RemoteManCLI.exe -target adstar -fw 0x20300000 uImage -q
RemoteManager(version 100)
        ….
start write(2507072 bytes)...
```

```
        100(2507072/2507072)
write :6.488229 sec
386403 bytes/sec

C:₩work>RemoteManCLI.exe -target adstar -fw 0x20a00000 ramdisk_adstar.img.gz -q
RemoteManager(version 100)
        ….
start write(332575 bytes)…
        100(332575/332575)
write :1.000947 sec
332260 bytes/sec
```

U-boot의 nand명령을 사용하여 NAND플래쉬 메모리의 각 영역을 erase한 후에 Write한다. 다음은 NAND플래쉬 메모리의 0x0 오프셋으로부터 2MB를 Erase한다.

```
adStar # nand erase 0x0 0x200000
NAND erase: device 0 offset 0x0, size 0x300000
Erasing at 0x1e0000 -- 100% complete.
OK
```

DRAM의 0x20300000주소에 있는 커널이미지를 NAND플래쉬의 0x0오프셋으로 2MB만큼 Write한다.

```
adStar # nand write.jffs2 0x20300000 0x0 0x200000
NAND write: device 0 offset 0x0, size 0x200000
 2097152 bytes written: OK
```

NAND플래쉬의 0x200000 오프셋으로부터 2MB를 Erase한다.

```
adStar # nand erase 0x200000 0x200000
NAND erase: device 0 offset 0x200000, size 0x200000
Erasing at 0x3e0000 -- 100% complete.
OK
```

DRAM의 0x20a00000주소에 있는 램디스크 이미지를 NAND플래쉬 메모리의 0x200000오프셋으로 2MB만큼 Write한다.

```
adStar # nand write.jffs2 0x20a00000 0x200000 0x200000
```

```
NAND write: device 0 offset 0x200000, size 0x200000
 2097152 bytes written: OK
```

adStar STK보드를 리셋하여 커널이 실행되는지 확인한다. 키 입력지연시간이 0이 될 때까지 키 입력이 없으면 Boot Command가 실행된다. Boot Command는 Serial Flash로부터 커널과 램디스크 이미지를 DRAM으로 복사한후 커널을 실행한다.

```
U-Boot 2010.12 (Apr 24 2012 - 11:14:25)

BOARD: ADSTAR
DRAM:  16 MiB
FLASH: 512 KiB
NAND:  128 MiB
In:    serial
Out:   serial
Err:   serial
Hit any key to stop autoboot:  0

NAND read: device 0 offset 0x0, size 0x200000
 2097152 bytes read: OK

NAND read: device 0 offset 0x200000, size 0x200000
 2097152 bytes read: OK
Boot reached stage 1
## Booting kernel from Legacy Image at 20300000 ...
Boot reached stage 2
Boot reached stage 3
   Image Name:   Linux-2.6.29-uc0
   Image Type:   AE32K Linux Kernel Image (gzip compressed)
   Data Size:    1208091 Bytes = 1.2 MiB
   Load Address: 20000000
   Entry Point:  20000000
   Verifying Checksum ... OK
Boot reached stage 4
Boot reached stage 5
Boot reached stage 6
Boot reached stage 14
   Uncompressing Kernel Image ... OK
```

```
Boot reached stage 7
Boot reached stage 8
Boot reached stage 15

Starting kernel ...

Start Kernel
[      0.000000] Linux version 2.6.29-uc0 (jman@swlinux) (/ Floating-point
optimized again / Double precision BUG Fixed)) #19
PREEMPT Tue Apr 24 16:02:02 KST 2012
[      0.000000] Machine: ADSTAR Board(Advanced Digital Chips inc.,
http://www.adc.co.kr)
...
```

Revision History of Document

Revision No.		
Ver 1.0	Initial release	04/23/2012